**HTML Lecture-1**.

**Points needs to be discussed before we start with actual concepts:**
What is Web Development?
What do you know about web development?
Examples,websites like e-commerce,edtech,stock market.
Web Tech Stack-HTML,CSS,JS,Tailwind CSS,bootstrap,React JS,NodeJS,Express JS,MongoDB

**Why frontend->backend->mongodb**
Frontend-do not stores the data
Backend stores data in data structures temporarily(reload krte hi gayab)
Database-Stores data permanently.

**Extensions needs to be taken:**
Boiler plate,prettier,Code runner(live server),code snippets,tag auto-rename,bracket pair closure.Bolier plate code ko ache se explain krna

**Installations:**
Vscode install thoda bhot kha se kre etc.

Then start with HTML Content.
Boiler Plate Code-minimal necessary code to start with

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
 </head>
<body>
    <header>
        <h1>Welcome to My Website</h1>
    </header>
    <main>
        <p>This is the main content area.</p>
    </main>
    <footer>
        <p>© 2024 My Website</p>
```

```
    </footer>
</body>
</html>
```

The boilerplate code in HTML refers to the minimal necessary code structure that you start with when creating a new HTML document. This provides a foundation and ensures that your HTML document adheres to the standards and best practices. Below is a typical HTML5 boilerplate code with explanations for each part:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="styles.css">
    <script src="script.js" defer></script>
</head>
<body>
    <header>
        <h1>Welcome to My Website</h1>
    </header>
    <main>
        <p>This is the main content area.</p>
    </main>
    <footer>
        <p>© 2024 My Website</p>
    </footer>
</body>
</html>
```

**Explanation:**

1. `<!DOCTYPE html>`:
   - Declares the document type and version of HTML. `<!DOCTYPE html>` tells the browser that the document is an HTML5 document.

2. `<html lang="en">`:
   - The `<html>` element is the root element of an HTML page. The `lang="en"` attribute specifies the language of the document (English in this case).

3. `<head>`:

- The `<head>` element contains meta-information about the HTML document, such as its title, character set, styles, and scripts.

4. `<meta charset="UTF-8">`:
   - This meta tag specifies the character encoding for the document. `UTF-8` is a standard character encoding that supports many languages and symbols.

5. `<meta name="viewport" content="width=device-width, initial-scale=1.0">`:
   - This meta tag ensures that the webpage is responsive and sets the viewport to match the device's width. It helps in making the website mobile-friendly.

6. `<title>Document</title>`:
   - The `<title>` element specifies the title of the document, which is displayed in the browser's title bar or tab.

7. `<link rel="stylesheet" href="styles.css">`:
   - The `<link>` element is used to link external stylesheets to the HTML document. In this case, it links to a CSS file named `styles.css`.

8. `<script src="script.js" defer></script>`:
   - The `<script>` element is used to include JavaScript files. The `defer` attribute ensures that the script is executed after the HTML document has been completely parsed.

9. `<body>`:
   - The `<body>` element contains the content of the HTML document that is displayed in the web browser.

10. `<header>`:
    - The `<header>` element represents the introductory content, typically containing a heading and other introductory content or navigation links.

11. `<h1>Welcome to My Website</h1>`:
    - The `<h1>` element defines the main heading of the document.

12. `<main>`:
    - The `<main>` element represents the dominant content of the `<body>` of the document. This content should be unique to the document and should not contain any content that is repeated across other pages (like sidebars or navigation links).

13. `<p>This is the main content area.</p>`:
    - The `<p>` element defines a paragraph of text.

14. `<footer>`:

- The `<footer>` element represents the footer of the document or section. It typically contains metadata about the document, such as the author, copyright information, or links to related documents.

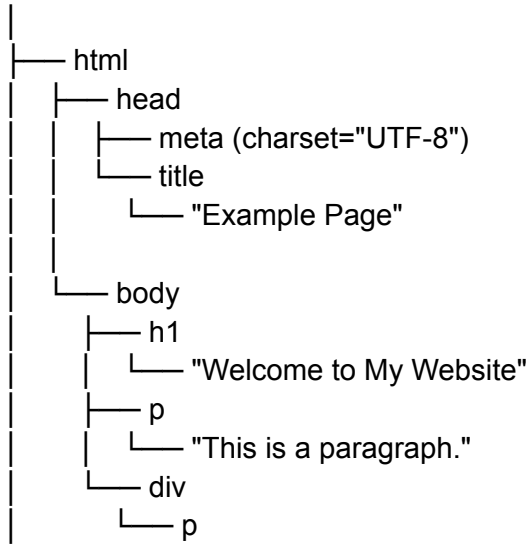
15. `<p>© 2024 My Website</p>`:
- Another `<p>` element within the `<footer>` contains the copyright information.

This boilerplate sets up a basic structure for an HTML document that you can build upon.

DOM-Document Object Module

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>Example Page</title>
</head>
<body>
   <h1>Welcome to My Website</h1>
   <p>This is a paragraph.</p>
   <div>
      <p>Nested paragraph inside a div.</p>
   </div>
</body>
</html>
```

HTML Tree Generator



```
|           └── "Nested paragraph inside a div."
```

**Block-Level Elements:**
**Tags:**
**1.Heading Tags:From H1 to H6:**
These are to give main headings,sub headings,sub titles etc. using different sets of headings.

**2.paragraph:**now that you have given headings and subheading you have to describe it or elaborate.

**3.div tag:** used to group together elements to apply styles or scripting, or to organize the layout of a webpage.

Example:

```
<div>
        <h2>Heading 1</h2>
        <p>This is the first paragraph. It provides some introductory information about the first topic.</p>
</div>
<div>
        <h2>Heading 2</h2>
        <p>This is the second paragraph. It provides some additional information about the second topic.</p>
</div>
```

**4.break tag:** gives a line break
<br> or  <br/>
Self-closing Tag

```
<p>
   This is the first line.<br>
   This is the second line.
</p>
```

**Handling Different Types of Tags:**

- **Opening Tags (e.g., `<div>`):** The browser creates a new node in the DOM.
- **Closing Tags (e.g., `</div>`):** The browser closes the current node and moves back to its parent node.
- **Self-Closing Tags (e.g., `<br>`, `<img />`):** The browser creates a node for the tag and immediately closes it.

**5.Img tag:** used to embed images into a webpage. It is an empty (self-closing) tag, meaning it does not have a closing tag and does not contain any content between opening and closing tags.

<img src="image-url" alt="Description of the image">

## Common Attributes

1. **`src`** (required):
   - Specifies the path to the image file.
   - Can be a relative URL (e.g., `"images/photo.jpg"`) or an absolute URL (e.g., `"https://example.com/photo.jpg"`).
2. **`alt`** (required):
   - Provides alternative text for the image if it cannot be displayed.
   - Important for accessibility and SEO. Descriptive alternative text helps screen readers convey the image content to users with visual impairments.
3. **`width`** (optional):
   - Specifies the width of the image in pixels or as a percentage of the containing element.
4. **`height`** (optional):
   - Specifies the height of the image in pixels or as a percentage of the containing element.

Img tag is an inline block means all images with come in same line but can be adjusted in terms of height and width.

Span tag is an inline element.

What happens if block element is placed inside inline?Will it become inline too?

**6.Anchor Tag:** Specifies the URL of the page the link goes to. It can be an absolute URL, a relative URL, or an anchor within the same page.

`_self` (default): Opens the link in the same frame as it was clicked.

`_blank`: Opens the link in a new window or tab.

<a href="https://www.example.com" target="_self">Open in New Tab</a>

<a href="https://www.example.com" target="_blank">Open in New Tab</a>

**Addtional:How to embed a vedio on a webpage?**

<iframe width="560" height="315"
src="https://www.youtube.com/embed/OjFxqCApM6E?si=phEtfEwxAqX7MpxR" title="YouTube video

player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" referrerpolicy="strict-origin-when-cross-origin" allowfullscreen></iframe>

**7.List:**

**Unordered List:**

An unordered list is a collection of items where the order does not matter. The items are typically displayed with bullet points.

### Attributes:

**`type`**: Specifies the type of marker to use (e.g.,disc,circle,square).

```
<ul>
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>
<ul>
  <li>Apple</li>
  <li>Banana</li>
  <li>Cherry</li>
</ul>
```

**Ordered List:**

An ordered list is a collection of items where the order does matter. The items are typically displayed with numbers or letters.

### Attributes:

**type**: Specifies the type of marker to use (e.g., `1`, `A`, `a`, `I`, `i`).

```
<ol type="A">
  <li>Step 1</li>
  <li>Step 2</li>
  <li>Step 3</li>
</ol>
```

**start:** Specifies the start value of the list.

```
<ol start="3">

  <li>Item 3</li>

  <li>Item 4</li>

  <li>Item 5</li>

</ol>
```

**Definition List:**

A definition list is a collection of terms and their corresponding descriptions.

```
<dl>

  <dt>HTML</dt>

  <dd>HyperText Markup Language</dd>

  <dt>CSS</dt>

  <dd>Cascading Style Sheets</dd>

  <dt>JavaScript</dt>

  <dd>Programming language of the web</dd>

</dl>
```

**Nested List:**

```
<ul>
  <li>Fruits
    <ul>
      <li>Apple</li>
      <li>Banana</li>
      <li>Cherry</li>
    </ul>
  </li>
  <li>Vegetables
    <ul>
      <li>Carrot</li>
      <li>Broccoli</li>
      <li>Spinach</li>
    </ul>
  </li>
</ul>
```

**Other Tags:**

**Bold Text**

**Bold Text**

**Italic Text**

**Italic Text**

**n<sup>2</sup>**

**H<sub>2</sub>O**

**How to add copyright symbol?**

```
<p>&copy; 2024 Your Company Name. All rights reserved.</p>

<p>&#169; 2024 Your Company Name. All rights reserved.</p>

<p>&#xA9; 2024 Your Company Name. All rights reserved.</p>
```

**Form in HTML:**

Common form elements include `<input>`, `<textarea>`, `<select>`, `<button>`, and `<label>`.

### Text Input

```
<label for="name">Name:</label>

<input type="text" id="name" name="name">
```

What if for and id doesn’t match?

### Password Input

```
<label for="password">Password:</label>

<input type="password" id="password" name="password">
```

### Radio Buttons

```
<label for="gender">Gender:</label>

<input type="radio" id="male" name="gender" value="male"> Male
```

```
<input type="radio" id="female" name="gender" value="female"> Female
```

**Checkboxes**

```
<label for="hobbies">Hobbies:</label>

<input type="checkbox" id="sports" name="hobbies" value="sports"> Sports


<input type="checkbox" id="music" name="hobbies" value="music"> Music
```

**Dropdown List**

```
<label for="country">Country:</label>

<select id="country" name="country">

  <option value="usa">USA</option>

  <option value="canada">Canada</option>

  <option value="mexico">Mexico</option>

</select>
```

**Text Area**

```
<label for="message">Message:</label>

<textarea id="message" name="message" rows="4" cols="50"></textarea>
```

**Submit Button**

```
<button type="submit">Submit</button>
```

## Semantic Tags

Semantic tags clearly describe their meaning in a human- and machine-readable way. They provide information about the structure and meaning of the content within them. This makes it easier for search engines and other user agents to understand the content of a webpage.

**Examples of Semantic Tags:**

- `<header>`: Represents introductory content or a set of navigational links.
- `<nav>`: Defines a set of navigation links.
- `<section>`: Defines a section in a document, often with its own heading.
- `<footer>`: Contains information about its containing element, like author information or copyright statements.

## Non-Semantic (Unsemantic) Tags

Non-semantic tags do not convey any meaning about their content. They are used purely for styling and layout purposes, without providing any context about the content they enclose.

**Examples of Non-Semantic Tags:**

- `<div>`: A generic container for grouping elements without any specific meaning.
- `<span>`: A generic inline container used for styling or grouping small sections of text.

## Text-Based Inputs

1. **`<input type="text">`**
    - **Purpose**: General single-line text input.
    - **Example**: Username, search queries.
2. **`<input type="password">`**
    - **Purpose**: Hides input text for sensitive information like passwords.
    - **Example**: User passwords.
3. **`<input type="email">`**
    - **Purpose**: Validates that the input is in the format of an email address.
    - **Example**: User email addresses.
4. **`<input type="url">`**

- **Purpose**: Validates that the input is in the format of a URL.
- **Example**: Website URLs.

## Date and Time Inputs

1. **<input type="date">**
   - **Purpose**: Provides a date picker.
   - **Example**: Birthdates, event dates.
2. **<input type="time">**
   - **Purpose**: Provides a time picker.
   - **Example**: Appointment times.
3. **<input type="datetime-local">**
   - **Purpose**: Allows the selection of a date and time.
   - **Example**: Scheduling events.